

# Complex Multiply-Add and Other Related Operators

Miloš D. Ercegovac<sup>a</sup> and Jean-Michel Muller<sup>b</sup>

<sup>a</sup>Computer Science Department, UCLA, Los Angeles, CA 90025, USA

<sup>b</sup>ENS-Lyon, France

## ABSTRACT

In this work we present algorithms and schemes for computing several common arithmetic expressions defined in the complex domain as hardware-implemented operators. The operators include Complex Multiply-Add ( $CMA : ab + c$ ), Complex Sum of Products ( $CSP : ab + ce + f$ ), Complex Sum of Squares ( $CSS : a^2 + b^2$ ), and Complex Integer Powers ( $CIPk : x^2, x^3, \dots, x^k$ ). The proposed approach is to map the expression to a system of linear equations, apply a complex-to-real transform, and compute the solutions to the linear system using a digit-by-digit, the most significant digit first, recurrence method. The components of the solution vector corresponds to the expressions being evaluated. The number of digit cycles is about  $m$  for  $m$ -digit precision. The basic modules are similar to left-to-right multipliers. The interconnections between the modules are digit-wide.

**Keywords:** Complex arithmetic, multiply-add, sum of products, sum of squares, integer powers

## 1. INTRODUCTION

With the exception of complex addition and multiplication,<sup>1,2,3</sup> complex online SVD,<sup>4</sup> complex operations are typically not implemented in hardware. Recently, hardware-oriented methods for complex division and square root have been introduced.<sup>5,6</sup>

In this paper we describe a new method for computing several common arithmetic expressions defined in the complex domain, suitable for hardware implementation as operators. The operators include Complex Multiply-Add ( $CMA : ab + c$ ), Complex Sum of Products ( $CSP : ab + ce + f$ ), Complex Sum of Squares ( $CSS : a^2 + b^2$ ), and Complex Integer Powers ( $CIPk : x^2, x^3, \dots, x^k$ ). The variables and results are fixed-point complex numbers. The proposed approach is to map the expression to a system of linear equations, apply a complex-to-real transform, and compute the solutions to the linear system using a digit-by-digit, the most significant digit first method. The components of the solution vector corresponds to the expressions being evaluated. The number of digit cycles is about  $m$  for  $m$ -digit precision. The basic modules are similar in complexity to left-to-right multipliers. The interconnections between the modules are digit-wide. The proposed method is a generalization of a polynomial evaluation method over the reals introduced as the E-method,<sup>7,8</sup> and recently discussed in.<sup>9</sup> This paper is based on the report<sup>10</sup> where the complex E-method is introduced and discussed in general terms.

The method uses the following approach: (i) an expression is mapped onto a system of linear equations, (ii) a transform is applied to change the complex domain to the real domain, and (iii) the system is solved in a digit-by-digit manner, the most-significant digit first.

We first review the transform which allows the method to be used in the complex field  $\mathbf{C}$  as discussed in.<sup>11</sup> Then we show how to use the complex evaluation method (CE-method) in implementing complex operators  $CMA$ ,  $CSP$ ,  $CSS$ , and  $CIP$ .

---

Further author information: (Send correspondence to M. Ercegovac, milos@cs.ucla.edu)

## 2. COMPLEX-REAL (CR) TRANSFORMS

Complex numbers can be represented by  $2 \times 2$  skew-symmetric matrices

$$x + iy \leftrightarrow \begin{pmatrix} x & -y \\ y & x \end{pmatrix} \quad (1)$$

This isomorphism holds for complex addition and multiplication which are used in the proposed method.

Consequently, an  $m \times n$  matrix of complex numbers can be represented as a  $2m \times 2n$  matrix of real numbers. For  $n \times n$  complex matrices, considered in this paper, the transform from the complex domain to the real domain is as follows.

The *CR-transform* of a  $n$ -dimensional *complex* linear system is a  $2n$ -dimensional *real* linear system. For example, let  $n = 3$ , then the CR transform is

$$\begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,1} & a_{3,2} & a_{3,3} \end{pmatrix} \times \begin{pmatrix} s_1 \\ s_2 \\ s_3 \end{pmatrix} = \begin{pmatrix} t_1 \\ t_2 \\ t_3 \end{pmatrix} \quad (2)$$

$$\begin{pmatrix} a_{1,1}^r & -a_{1,1}^i & a_{1,2}^r & -a_{1,2}^i & a_{1,3}^r & -a_{1,3}^i \\ a_{1,1}^i & a_{1,1}^r & a_{1,2}^i & a_{1,2}^r & a_{1,3}^i & a_{1,3}^r \\ a_{2,1}^r & -a_{2,1}^i & a_{2,2}^r & -a_{2,2}^i & a_{2,3}^r & -a_{2,3}^i \\ a_{2,1}^i & a_{2,1}^r & a_{2,2}^i & a_{2,2}^r & a_{2,3}^i & a_{2,3}^r \\ a_{3,1}^r & -a_{3,1}^i & a_{3,2}^r & -a_{3,2}^i & a_{3,3}^r & -a_{3,3}^i \\ a_{3,1}^i & a_{3,1}^r & a_{3,2}^i & a_{3,2}^r & a_{3,3}^i & a_{3,3}^r \end{pmatrix} \times \begin{pmatrix} s_1^r \\ s_1^i \\ s_2^r \\ s_2^i \\ s_3^r \\ s_3^i \end{pmatrix} = \begin{pmatrix} t_1^r \\ t_1^i \\ t_2^r \\ t_2^i \\ t_3^r \\ t_3^i \end{pmatrix} \quad (3)$$

where  $a_{j,k} = a_{j,k}^r + ia_{j,k}^i$ ,  $s_j = s_j^r + is_j^i$  and  $t_j = t_j^r + it_j^i$ . These two linear systems are *equivalent*.

The real linear system (??) is obtained from the complex linear system (??) by replacing each complex element by the  $2 \times 2$  matrix defined in (??). In the following sections we review a hardware-oriented method for solving such a system.<sup>?</sup>

## 3. REAL E-METHOD

For simplicity, we discuss here radix-2 E-method. Adaptation to higher radices is straightforward. The radix-2 method consists in solving the  $n$ -dimensional linear system

$$As = v$$

using the following iteration on residuals:

$$w^{(j)} = 2 \times [w^{(j-1)} - Ad^{(j-1)}] \quad (4)$$

with  $w^{(0)} = [v_0, v_1, \dots, v_n]^T$ , and  $d^{(j)} = [d_0, d_1, \dots, d_n]^T$  where the digits  $d_k^{(j)}$  are in  $\{-1, 0, 1\}$ . Define the number  $D_k^{(j)} = d_k^{(0)} \cdot d_k^{(1)} d_k^{(2)} \dots d_k^{(j)}$  (the  $d_k^{(j)}$  are the digits of a radix-2 signed-digit representation of  $D_k^{(j)}$ ). By induction,

$$w^{(j)} = 2^j [w^{(0)} - AD^{(j-1)}]. \quad (5)$$

Using (??), one can show that if the residuals  $|w_k^{(j)}|$  are bounded, then for all  $k$ ,  $D_k^{(j)}$  converges to  $s_k$  as  $j$  goes to infinity. At step  $j$  we must select a value of the digits  $d_k^{(j)}$  from the residuals  $w_k^{(j)}$  such that the values  $w_k^{(j+1)}$  remain bounded. The following selection function, proposed in <sup>?</sup> as a form of rounding, achieves such a choice.

$$SEL(x) = \begin{cases} \text{sign } x \times \lfloor |x + 1/2| \rfloor, & \text{if } |x| \leq 1 \\ \text{sign } x \times \lfloor |x| \rfloor, & \text{otherwise,} \end{cases} \quad (6)$$

To avoid carry-propagate addition in the recurrence, the selection function is applied to an estimate of the residual :  $d_k^{(j)} = SEL(\hat{w}_k^{(j)})$ , where  $\hat{w}_k^{(j)}$  is a low-precision approximation to  $w_k^{(j)}$ .

The iterations converge to the desired result if residual vector  $w^{(j)}$  is bounded. Let  $\xi$ ,  $\alpha$  and  $\Delta$  (with  $0 \leq \Delta < 1$ ) be constants such that

1. Sum of magnitudes of off-diagonal elements in matrix  $A$ :  $\leq \alpha$ ;
2. Magnitudes of right-hand side elements:  $\leq \xi$ ;
3.  $|w_{k,r}^{(j)} - \hat{w}_{k,r}^{(j)}| \leq \frac{\Delta}{2}$ , and  $|w_{k,i}^{(j)} - \hat{w}_{k,i}^{(j)}| \leq \frac{\Delta}{2}$

Since  $|d_{k,r}^{(j-1)} - \hat{w}_{k,r}^{(j-1)}| \leq 1/2$  and  $|d_{k,i}^{(j-1)} - \hat{w}_{k,i}^{(j-1)}| \leq 1/2$ , from (??) we find

$$|w_{k,r}^{(j)}| \leq 2 \left( \frac{1}{2} + \frac{\Delta}{2} + \alpha \right) = 1 + \Delta + 2\alpha. \quad (7)$$

The same bound holds for  $|w_{k,i}^{(j)}|$ . For this bound to be feasible, we must assure that a suitable choice of  $d_{k,r}^{(j)}$  and  $d_{k,i}^{(j)}$  in  $\{-1, 0, 1\}$  is possible. This requires that  $|w_{k,r}^{(j)}|$  and  $|w_{k,i}^{(j)}|$  should be less than  $3/2$ . Therefore,

$$\Delta + 2\alpha \leq \frac{1}{2} \quad (8)$$

Since  $|w_{k,r}^{(0)}|$  and  $|w_{k,i}^{(0)}|$  must also be less than  $3/2$ , we get

$$\xi \leq \frac{3}{2} \quad (9)$$

In the complex E-method the coefficient matrix  $A$ , the solution vector  $s$  and the right-hand side  $v$  are in the complex domain. Moreover, the complex residual vector is denoted as

$$w^{(j)} = [w_{0,r}^{(j)}, w_{0,i}^{(j)}, w_{1,r}^{(j)}, w_{1,i}^{(j)} \dots, w_{n,r}^{(j)}, w_{n,i}^{(j)}]$$

and the complex digit vector  $d^{(j)}$  is written as

$$d^{(j)} = [d_{0,r}^{(j)}, d_{0,i}^{(j)}, d_{1,r}^{(j)}, d_{1,i}^{(j)}, \dots, d_{n,r}^{(j)}, d_{n,i}^{(j)}]$$

The mapping of operators on linear systems and complex residual iterations are discussed in the following sections for each complex operator.

#### 4. COMPLEX MULTIPLY-ADD OPERATOR (CMA)

The operator computes  $y = ab + c$ . In the real domain, the mapping to a linear system is

$$\begin{pmatrix} 1 & -a \\ 0 & 1 \end{pmatrix} \times \begin{pmatrix} s_0 \\ s_1 \end{pmatrix} = \begin{pmatrix} c \\ b \end{pmatrix}$$

where the solution is obtained as  $s_0 = y$ . In the complex domain, the variables are  $y = y^r + iy^i$ ,  $a = a^r + ia^i$ ,  $b = b^r + ib^i$ , and  $c = c^r + ic^i$ . The mapping in this case is

$$\begin{pmatrix} 1 & 0 & -a^r & a^i \\ 0 & 1 & -a^i & -a^r \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \times \begin{pmatrix} s_0^r \\ s_0^i \\ s_1^r \\ s_1^i \end{pmatrix} = \begin{pmatrix} c^r \\ c^i \\ b^r \\ b^i \end{pmatrix}$$

so that  $s_0^r = y^r$  and  $s_0^i = y^i$ .

The residual recurrences are:

$$\begin{aligned}
w_{0,r}^{(j)} &= 2 \left[ w_{0,r}^{(j-1)} - d_{0,r}^{(j-1)} + a^r d_{1,r}^{(j-1)} - a^i d_{1,i}^{(j-1)} \right] \\
w_{0,i}^{(j)} &= 2 \left[ w_{0,i}^{(j-1)} - d_{0,i}^{(j-1)} + a^i d_{1,r}^{(j-1)} + a^r d_{1,i}^{(j-1)} \right] \\
w_{1,r}^{(j)} &= 2 \left[ w_{1,r}^{(j-1)} - d_{1,r}^{(j-1)} \right] \\
w_{1,i}^{(j)} &= 2 \left[ w_{1,i}^{(j-1)} - d_{1,i}^{(j-1)} \right]
\end{aligned} \tag{10}$$

with the initial conditions

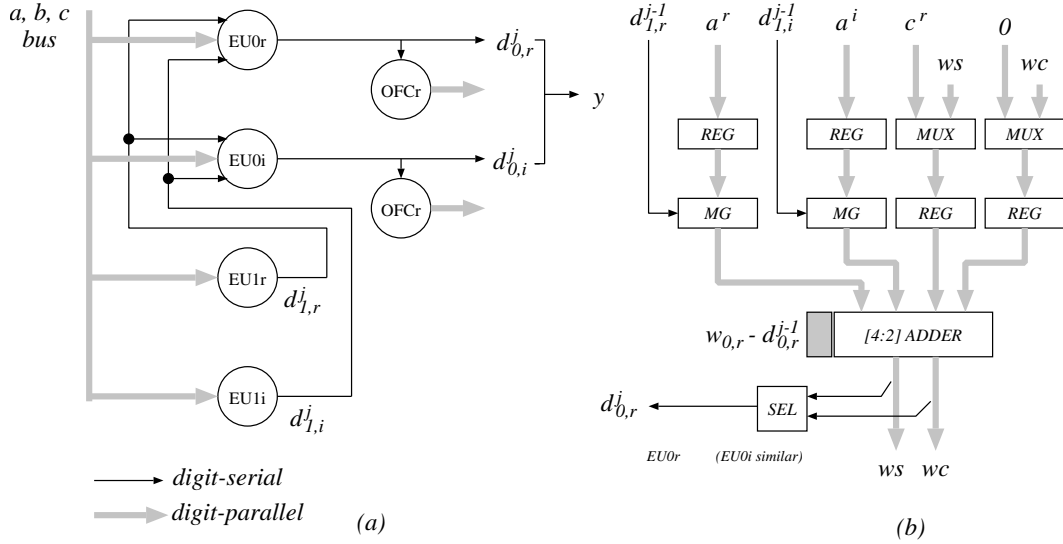
$$w_{0,r}^{(0)} = c^r, \quad w_{0,i}^{(0)} = c^i, \quad w_{1,r}^{(0)} = b^r, \quad w_{1,i}^{(0)} = b^i$$

The convergence requires that the following conditions are satisfied

$$|a^r| + |a^i| \leq \alpha, \quad |c^r|, |c^i|, |b^r|, |b^i| \leq 3/2 \tag{11}$$

From condition (??) and using  $\Delta = 1/8$ , we get  $\alpha \leq 3/16$ . Therefore,  $|a^r|, |a^i| \leq 3/32$  assures convergence of the algorithm. This range reduction of the input  $a$  can be achieved by scaling of the initial value.?

A scheme for implementing the *CMA* operator is shown in Figure ??(a) and the corresponding elementary unit (*EU0*) is illustrated in Figure ??(b). Elementary units *EU1* can be simplified as discussed below. A bit-parallel bus transmits  $a$  values in a broadcast mode, while  $b$  and  $c$  variables are loaded in separate cycles. Note that the initialization cycles could be shorter than the iteration cycles.



**Figure 1.** (a) Scheme for *CMA* operator. (b) Block diagram of elementary unit.

A block diagram of an elementary unit *EU0* (real part only) for *CMA* operator is shown in Figure ??(b). It uses the following modules:

- Registers (4)
- Multiple generators  $MG$  (2), producing  $\{-1, 0, 1\} \times a^r$  and  $\{-1, 0, 1\} \times a^i$ , and buffers

- Multiplexers MUX (2) for initializing the residual
- A [4:2] adder (the shaded MS part performs the indicated subtraction of the selected digit)
- Output digit selection  $SEL$  (a small table or a gate network)

The elementary unit  $EU1$  can be greatly simplified. This module, in general radix- $r$  case is effectively a recoder from the digit set  $\{0, 1, \dots, r-1\}$  to the the set  $\{-a, \dots, a\}$  where  $r/2 \leq a \leq r-1$ . In the case of radix-2, this recoding is unnecessary: the module is a left shift register.

The digit-serial outputs of  $EUs$  can be converted into digit-parallel form using on-the-fly converters  $OFCr$  and  $OFCi$  as indicated by the thick lines.<sup>?</sup>

The cycle time (without interconnect delay between units), in terms of a full adder delay  $t$ , is estimated as

$$\begin{aligned} T_{EU-CMA} &= t_{BUFF} + t_{MG} + t_{SEL} + t_{[4:2]} + t_{REG} \\ &\approx (0.4 + 0.3 + 1 + 1.3 + 0.9)t = 3.9t \end{aligned} \quad (12)$$

The cost, again in terms of area of a full adder  $A_{FA}$ , is estimated as

$$\begin{aligned} A_{EU-CMA}(m) &= A_{SEL} + 2A_{BUFF} + (m+2)[2A_{MG} \\ &\quad + 2A_{MUX} + A_{[4:2]} + 4A_{REG} + A_{OFC}] \\ &\approx [5 + 2 \times 0.4 + (m+2)(4 \times 0.45 \\ &\quad + 2.3 + 4 \times 0.6 + 2.1)]A_{FA} \\ &\approx (23 + 9m)A_{FA} \end{aligned} \quad (13)$$

The cost is estimated as area occupied by modules using the area of a full-adder  $A_{FA}$  as the unit. The areas of primitive modules are: Register  $A_{REG} = 0.6A_{FA}$ ; buffer  $A_{BUFF} = 0.4A_{FA}$ ; MUX  $A_{MUX} = 0.45A_{FA}$ ; multiple generator MG  $A_{MG} = 0.45A_{FA}$ ; [4:2] adder  $A_{[4:2]} = 2.3A_{FA}$ ; SEL  $A_{SEL} = 5A_{FA}$ , and on-the-fly converters  $A_{OFC} = 2A_{MUX} + 2A_{REG} = 2.1A_{FA}$ . A total cost of an  $m$ -bit  $CMA$  operator is

$$A_{CMA}(m) = 2 \times A_{EU-CMA}(m) + 2 \times (m+2)A_{REG} \approx (50 + 20m)A_{FA}$$

## 5. COMPLEX SUM OF PRODUCTS OPERATOR (CSP)

The operator computes  $y = ab + ce + f$ . In the real domain, the mapping to a linear system is

$$\begin{pmatrix} 1 & -a & -c \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \times \begin{pmatrix} s_0 \\ s_1 \\ s_2 \end{pmatrix} = \begin{pmatrix} f \\ b \\ e \end{pmatrix}$$

and the solution  $s_0 = y$ . In the complex domain,  $a = a^r + ia^i$ ,  $b = b^r + ib^i$ , etc. The mapping in this case is

$$\begin{pmatrix} 1 & 0 & -a^r & a^i & -c^r & c^i \\ 0 & 1 & -a^i & -a^r & -c^i & -c^r \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \times \begin{pmatrix} s_0^r \\ s_0^i \\ s_1^r \\ s_1^i \\ s_2^r \\ s_2^i \end{pmatrix} = \begin{pmatrix} f^r \\ f^i \\ b^r \\ b^i \\ e^r \\ e^i \end{pmatrix}$$

The residual recurrences are:

$$\begin{aligned}
w_{0,r}^{(j)} &= 2 \left[ w_{0,r}^{(j-1)} - d_{0,r}^{(j-1)} + a^r d_{1,r}^{(j-1)} - a^i d_{1,i}^{(j-1)} + c^r d_{2,r}^{(j-1)} - c^i d_{2,i}^{(j-1)} \right] \\
w_{0,i}^{(j)} &= 2 \left[ w_{0,i}^{(j-1)} - d_{0,i}^{(j-1)} + a^i d_{1,r}^{(j-1)} + a^r d_{1,i}^{(j-1)} + c^i d_{2,r}^{(j-1)} + c^r d_{2,i}^{(j-1)} \right] \\
w_{1,r}^{(j)} &= 2 \left[ w_{1,r}^{(j-1)} - d_{1,r}^{(j-1)} \right] \\
w_{1,i}^{(j)} &= 2 \left[ w_{1,i}^{(j-1)} - d_{1,i}^{(j-1)} \right] \\
w_{2,r}^{(j)} &= 2 \left[ w_{2,r}^{(j-1)} - d_{2,r}^{(j-1)} \right] \\
w_{2,i}^{(j)} &= 2 \left[ w_{2,i}^{(j-1)} - d_{2,i}^{(j-1)} \right]
\end{aligned} \tag{14}$$

with the initial conditions

$$w_{0,r}^{(0)} = f^r, \quad w_{0,i}^{(0)} = f^i, \quad w_{1,r}^{(0)} = b^r, \quad w_{1,i}^{(0)} = b^i, \quad w_{2,r}^{(0)} = e^r, \quad w_{2,i}^{(0)} = e^i$$

The convergence requires that the following conditions are satisfied

$$|a^r| + |a^i| + |c^r| + |c^i| \leq \alpha, \quad |f^r|, |f^i|, |b^r|, |b^i|, |e^r|, |e^i| \leq 3/2 \tag{15}$$

From condition (??) and using  $\Delta = 1/8$ , we get  $\alpha \leq 3/16$  and  $|a^r|, |a^i|, |c^r|, |c^i| \leq 3/64$  to assure convergence of the algorithm. This range reduction can be achieved by scaling of the initial values.<sup>?</sup>

A scheme for implementing the *CSP* operator is shown in Figure ??(a) and the corresponding elementary unit (*EU*) is illustrated in Figure ??(b). A bit-parallel bus transmits  $a$  and  $c$ , while the real and imaginary parts of  $f$ ,  $b$  and  $e$  are loaded in separate cycles. Note that the initialization cycles could be shorter than the iteration cycles.

A block diagram of elementary unit *EU0r* (real part only) for the *CSP* operator is shown in Figure ??(b). The modules used are:

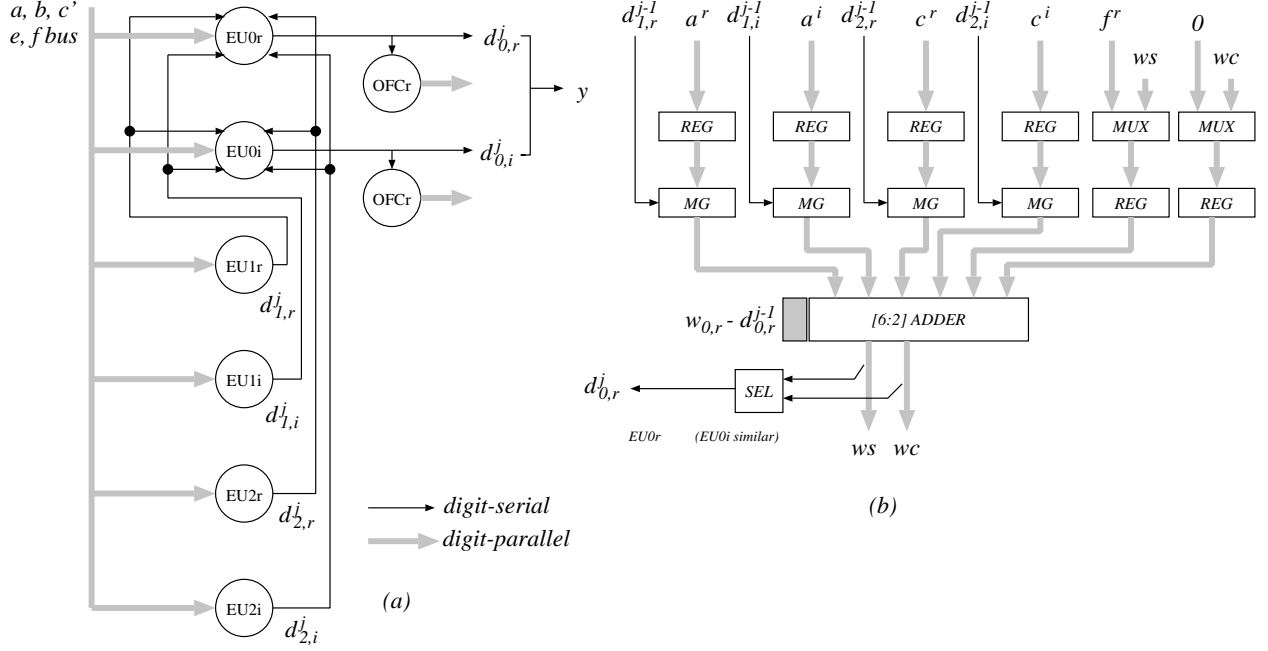
- Registers (6)
- Multiple generators MG (4), producing  $\{-1, 0, 1\} \times a^r$  and  $\{-1, 0, 1\} \times a^i$ , and buffers
- Multiplexers MUX (2) for initializing the residual
- A [6:2] adder (the shaded MS part performs the indicated subtraction of the selected digit)
- Output digit selection *SEL* (a small table or a gate network)

As in the *CMA* operator, the digit-serial outputs of the *EU0* can be converted into digit-parallel form using on-the-fly converters *OFCr* and *OFCi*. The cycle time, in terms of a full adder (complex gate) delay  $t$ , is estimated as

$$\begin{aligned}
T_{EU-CSP} &= t_{BUFF} + t_{MG} + t_{SEL} + t_{[6:2]} + t_{REG} \\
&\approx (0.4 + 0.3 + 1 + 2.3 + 0.9)t = 4.9t
\end{aligned} \tag{16}$$

The cost, again in terms of area of a full adder  $A_{FA}$ , is estimated as CHANGE

$$\begin{aligned}
A_{EU-CSP}(m) &= A_{SEL} + 4A_{BUFF} + (m+2)[4A_{MG} \\
&\quad + A_{MUX} + A_{[6:2]} + 6A_{REG} + 2A_{OFC}] \\
&\approx [5 + 4 \times 0.4 + (m+2)(5 \times 0.45 \\
&\quad + 4.3 + 6 \times 0.6 + 2 \times 2.1)]A_{FA} \\
&\approx (35 + 14m)A_{FA}
\end{aligned} \tag{17}$$



**Figure 2.** (a) Overall scheme for *CSP* operator. (b) Block diagram of elementary unit.

As discussed in the previous section, the cost is estimated as the area occupied by the modules using the area of a full-adder  $A_{FA}$  as the unit. A total cost of an  $m$ -bit *CSP* operator is

$$A_{CSP}(m) = 2 \times A_{EU-CSP}(m) + 4 \times (m + 2)A_{REG} \approx (70 + 30m)A_{FA}$$

## 6. COMPLEX SUM OF SQUARES OPERATOR (*CSS*)

This operator computes  $y = a^2 + b^2$ . In the real domain, the mapping to a linear system is

$$\begin{pmatrix} 1 & -a & -b \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \times \begin{pmatrix} s_0 \\ s_1 \\ s_2 \end{pmatrix} = \begin{pmatrix} 0 \\ a \\ b \end{pmatrix}$$

and the solution  $s_0 = y$ . In the complex domain,  $a = a^r + ia^i$ , and  $b = b^r + ib^i$ . The mapping in this case is

$$\begin{pmatrix} 1 & 0 & -a^r & a^i & -b^r & b^i \\ 0 & 1 & -a^i & -a^r & -b^i & -b^r \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \times \begin{pmatrix} s_0^r \\ s_0^i \\ s_1^r \\ s_1^i \\ s_2^r \\ s_2^i \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ a^r \\ a^i \\ b^r \\ b^i \end{pmatrix}$$

The residual recurrences are:

$$\begin{aligned}
w_{0,r}^{(j)} &= 2 \left[ w_{0,r}^{(j-1)} - d_{0,r}^{(j-1)} + a^r d_{1,r}^{(j-1)} - a^i d_{1,i}^{(j-1)} + b^r d_{2,r}^{(j-1)} - b^i d_{2,i}^{(j-1)} \right] \\
w_{0,i}^{(j)} &= 2 \left[ w_{0,i}^{(j-1)} - d_{0,i}^{(j-1)} + a^i d_{1,r}^{(j-1)} + a^r d_{1,i}^{(j-1)} + b^i d_{2,r}^{(j-1)} + b^r d_{2,i}^{(j-1)} \right] \\
w_{1,r}^{(j)} &= 2 \left[ w_{1,r}^{(j-1)} - d_{1,r}^{(j-1)} \right] \\
w_{1,i}^{(j)} &= 2 \left[ w_{1,i}^{(j-1)} - d_{1,i}^{(j-1)} \right] \\
w_{2,r}^{(j)} &= 2 \left[ w_{2,r}^{(j-1)} - d_{2,r}^{(j-1)} \right] \\
w_{2,i}^{(j)} &= 2 \left[ w_{2,i}^{(j-1)} - d_{2,i}^{(j-1)} \right]
\end{aligned} \tag{18}$$

with the initial conditions

$$w_{0,r}^{(0)} = 0, \quad w_{0,i}^{(0)} = 0, \quad w_{1,r}^{(0)} = a^r, \quad w_{1,i}^{(0)} = a^i, \quad w_{2,r}^{(0)} = b^r, \quad w_{2,i}^{(0)} = b^i$$

The convergence requires that the following conditions are satisfied

$$|a^r| + |a^i| + |b^r| + |b^i| \leq \alpha \tag{19}$$

From condition (??) and using  $\Delta = 1/8$ , we get  $\alpha \leq 3/16$  and  $|a^r|, |a^i|, |b^r|, |b^i| \leq 3/64$  to assure convergence of the algorithm. This range reduction can be achieved by scaling of the initial values.<sup>?</sup>

A scheme for implementing the *CSS* operator (general and elementary unit) is similar to that of Figure ?? . Consequently the delays and the cost are similar as estimated for the *CSP* operator.

## 7. COMPLEX INTEGER POWERS OPERATOR (*CIP*)

The operator computes in the real domain consecutive integer powers of the argument  $x$  in parallel:  $x^2, x^3, \dots, x^k$ . The corresponding linear system is

$$\begin{pmatrix} 1 & -x & 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & -x & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & 1 & -x \\ 0 & 0 & 0 & \cdots & 0 & 0 & 1 \end{pmatrix} \times \begin{pmatrix} s_0 \\ s_1 \\ \vdots \\ s_{k-1} \\ s_k \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \\ x \end{pmatrix}$$

and the integer powers are obtained as

$$s_0 = x^k, \quad s_1 = x^{k-1}, \dots, s_{n-1} = x^2$$

The mapping in the complex domain is shown next. The complex argument is  $z = x + iy$ .



$$A = \begin{pmatrix} 1 & 0 & -x & y & 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 1 & -y & -x & 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 1 & 0 & -x & y & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & 1 & -y & -x & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & 0 & 0 & 0 & 1 & 0 & -x & y \\ 0 & 0 & \dots & 0 & 0 & 0 & 0 & 1 & -y & -x \\ 0 & 0 & \dots & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & \dots & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

The components of the solution  $s$  of the linear system

$$\begin{aligned} A \times (s_0^r, s_0^i, s_1^r, s_1^i, \dots, s_{k-1}^r, s_{k-1}^i, s_k^r, s_k^i)^T \\ = (0, 0, 0, 0, \dots, 0, 0, x, y)^T \end{aligned} \quad (20)$$

are equal to the integer powers of  $z$ .

The residual recurrences are

$$\begin{aligned} w_{h,r}^{(j)} &= 2 \left[ w_{h,r}^{(j-1)} - d_{h,r}^{(j-1)} + x d_{h+1,r}^{(j-1)} - y d_{h+1,i}^{(j-1)} \right] \\ w_{h,i}^{(j)} &= 2 \left[ w_{h,i}^{(j-1)} - d_{h,i}^{(j-1)} + y d_{h+1,r}^{(j-1)} + x d_{h+1,i}^{(j-1)} \right] \end{aligned} \quad (21)$$

for  $h = k$ ,

$$\begin{aligned} w_{k,r}^{(j)} &= 2 \left[ w_{k,r}^{(j-1)} - d_{k,r}^{(j-1)} \right] \\ w_{k,i}^{(j)} &= 2 \left[ w_{k,i}^{(j-1)} - d_{k,i}^{(j-1)} \right] \end{aligned}$$

with the initial conditions for  $h = 0, \dots, k-1$

$$w_{h,r}^{(0)} = 0, \quad w_{h,i}^{(0)} = 0$$

and

$$w_{k,r}^{(0)} = x, \quad w_{k,i}^{(0)} = y$$

The convergence requires that the following conditions are satisfied

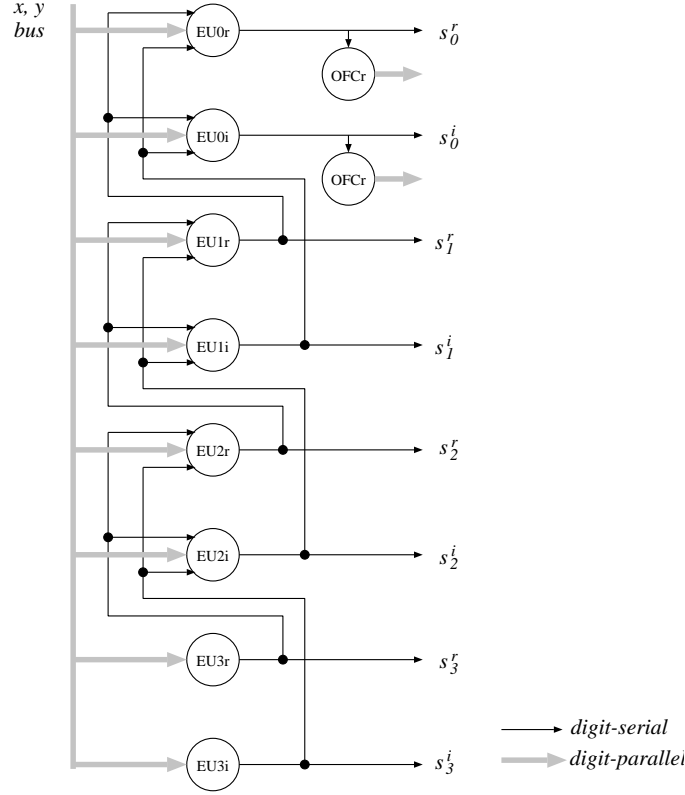
$$|x| + |y| \leq \alpha \quad (22)$$

From condition (??) and using  $\Delta = 1/8$ , we get  $\alpha \leq 3/16$  and  $|x|, |y| \leq 3/32$  to assure convergence of the algorithm. This range reduction can be achieved by scaling of the initial values.<sup>?</sup>

A scheme for implementing the *CIP* operator is shown in Figure ???. The corresponding elementary unit is similar to the *EU* of the *CMA* operator, illustrated in Figure ??(b), with the same cycle time and cost. A bit-parallel bus transmits  $x$  and  $y$  values in a broadcast mode as discussed earlier. A total cost of an  $m$ -bit *CIP* $k$  operator is

$$A_{CIPk}(m) = (k-1) \times A_{EU-CMA}(m) + 2 \times (m+2)A_{REG}$$

The same scheme, with different initialization, can be used to evaluate complex polynomials.<sup>?</sup>



**Figure 3.** Scheme for *CIP* operator.

## 8. SUMMARY

We presented a general method for computing several commonly used arithmetic expressions in the complex domain: multiply-add, sum of products, sum of squares, and integer powers. The method consists of mapping the operators on diagonally-dominant systems of complex linear equations, transforming the system from the complex to the real domain, and solving it using digit-by-digit MSDF algorithm. The latency is roughly  $m$  cycles for  $m$  bits of precision and independent of the order of the resulting linear system. The cycle time is independent of  $m$ . We discussed the mapping of operators to linear systems, transforms from the real to the complex domains, the recurrences and convergence conditions. Implementations of the proposed operators are discussed at a high level with estimates of the cost and cycle time. The method used here has been applied in the case of complex polynomials and rational functions.?,?

## REFERENCES

1. T. Aoki, H. Amada, and T. Higuchi. Real/complex reconfigurable arithmetic using redundant complex number systems. *Proc. 13th IEEE Symposium on Computer Arithmetic*, pp.200-207, 1997.
2. M.D. Ercegovac. *A general method for evaluation of functions and computation in a digital computer*. PhD thesis, Dept. of Computer Science, University of Illinois, Urbana-Champaign, 1975.
3. M.D. Ercegovac. A General Hardware-oriented Method for Evaluation of Functions and Computations in a Digital Computer. *IEEE Trans. Comp.*, C-26(7):667–680, 1977.
4. M.D. Ercegovac and T. Lang, *Digital Arithmetic*, Morgan Kaufmann Publishers, San Francisco, 2004.
5. M.D. Ercegovac and J.-M. Muller. Complex Division with Prescaling of Operands. *IEEE International Conference on Application-Specific Systems, Architectures and Processors*, pp. 293-303, 2003.

6. M.D. Ercegovic and J.-M. Muller, Design of a complex divider. *Proc. SPIE on Advanced Signal Processing Algorithms, Architectures, and Implementations XII*, pp. 51-59, 2004.
7. M.D. Ercegovic and J.-M. Muller. Complex Square Root with Operand Prescaling. *IEEE International Conference on Application-Specific Systems, Architectures and Processors*, pp. 293-303, 2004.
8. M.D. Ercegovic and J.-M. Muller, Solving Systems of Linear Equations in Complex Domain : Complex E-Method. LIP Report No. 2007-2, École Normale Supérieure de Lyon, France.
9. M.D. Ercegovic and J.-M. Muller, A Hardware-Oriented Method for Evaluating Complex Polynomials. *IEEE International Conference on Application-Specific Systems, Architectures and Processors*, 2007.
10. R.D. McIlhenny, *Complex Number On-line Arithmetic for Reconfigurable Hardware: Algorithms, Implementations, and Applications*, Ph.D. Dissertation, Computer Science Department, University of California, 2002.
11. V. Oklobdzija, D. Villeger and T. Soulas, An Integrated Multiplier for Complex Numbers. *J. of VLSI Signal Processing*, vol.7, no. 3, pp.213-222, May 1994.
12. B.W.Y. Wei, H. Du, and H. Chen, A Complex-Number Multiplier Using Radix-4 Digits. *Proc. 12th IEEE Symposium on Computer Arithmetic*, pp. 84-90, 1995